CS448 PSO

Week 4

CS448 staff

3NF & BCNF

- FD: X->A is “Nontrivial” = right-side attribute not in left side
- 3NF : A relation R is in 3NF iff (if and only if)

  for every nontrivial FD: X->A, either:

  - X is a superkey, or
  - A is prime = member of at least one key.
- A relation R is in BCNF if for every nontrivial FD for R, say X->A,
  - X is a superkey
  - ~~A is prime = member of at least one key.~~

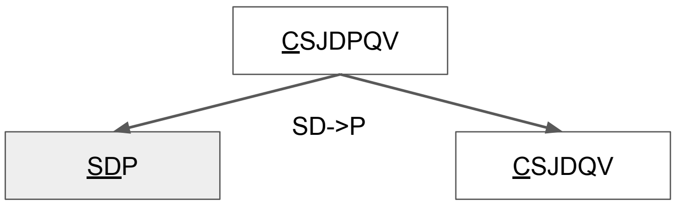
Example 1: decompose a schema into BCNF

- Schema: CSJDPQV
- FDs: {SD->P, J->S, JP->C}
- FDs violate BCNF

Left sides SD, J, JP are not superkey

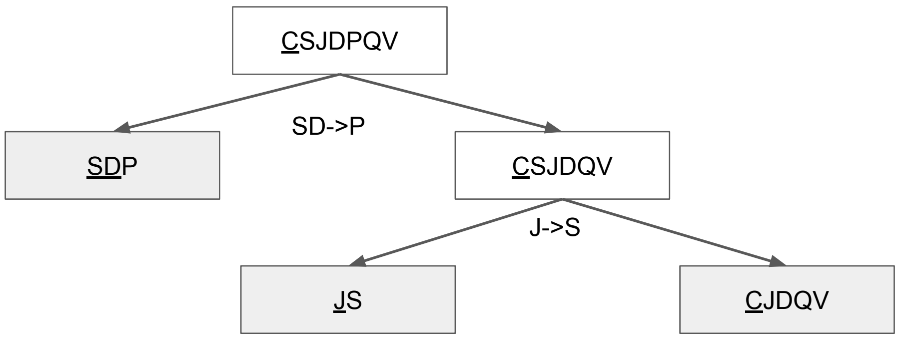
Example 1: decompose a schema into BCNF

- Schema: CSJDPQV
- FDs: {SD->P, J->S, JP->C}
- FDs violate BCNF



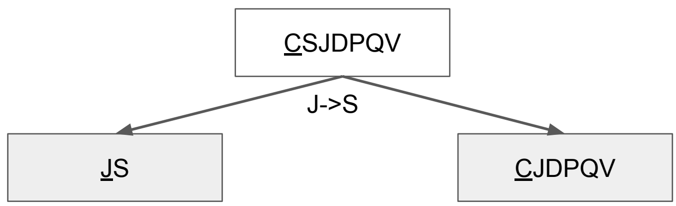
Example 1: decompose a schema into BCNF

- Schema: CSJDPQV
- FDs: {SD->P, J->S, JP->C}
- FDs violate BCNF



Alternatives in decomposing into BCNF

- Schema: CSJDPQV
- FDs: {SD->P, J->S, JP->C}
- FDs violate BCNF

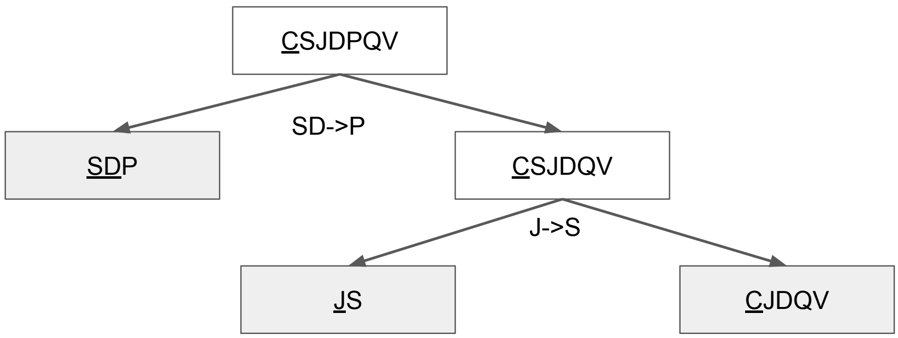


Which alternative should be used?

- Choose the alternatives based on the semantics of the application.
- Example: R=(course id, course name, course abbreviation, year, instructor)
  - course abbreviation->course name
  - course name, year->instructor
- The most frequently used query is selecting instructors given the course name and year.
- Two decompositions:
  - (course name, course abbreviation) and (course id, course abbreviation, year, instructor)
  - (course name, year, instructor) and (course id, course name, course abbreviation, year)

Example 2: dependency-preserving decomposition into 3NF

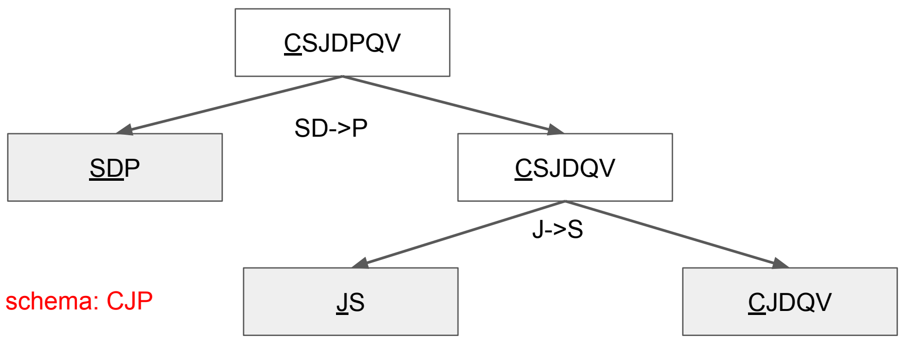
- Schema: CSJDPQV
- FDs: {SD->P, J->S, JP->C}
- SD->P, J->S violate 3NF



Left sides SD, J are not superkey
Right sides P, S are not *prime*

Example 2: dependency-preserving decomposition into 3NF

- Schema: CSJDPQV
- FDs: {SD->P, J->S, JP->C}
- SD->P, J->S violate 3NF



Add a relation schema: CJP

JP->C is not preserved in the decomposed relation.

Create views to consolidate a non-preserved FD

- Use materialized views to consolidate a non-preserved FD into one table
- Check the FD in that materialized view by making LHS of the FD the key for the view
- But will need to maintain the views when the base tables get updated

3NF synthesis

Take all the attributes over the original relation R and a minimal cover F for the FDs that hold over it and add a relation schema XA to the decomposition of R for each FD X->A in F.

If no decomposed table contains a candidate key for R, add a relation schema of any candidate key for R.

Minimal cover

Minimal (Canonical) Cover for a set F of FDs is a set G of FDs such that:

1. Every dependency in G is of the form X->A, where A is a single attribute
2. The closure F^+ is equal to the closure G^+
3. If we obtain a set H of dependencies from G by deleting one or more dependencies or by deleting attributes from a dependency in G, then $F^+ \neq H^+$

Example 3: find the minimal cover set

- A->B, ABCD->E, EF->G, EF->H, ACDF->EG

1. Rewrite ACDF->EG to ACDF->E, ACDF->G
2. ACDF->G is redundant
3. ACDF->E is redundant
4. Minimize left side of ABCD->E to ACD->E

- A->B, ACD->E, EF->G, EF->H

ACDF->G is redundant: A->B, ABCD->E => ACD->E => ACDF->EF->G => ACDF->G
ACDF->E is redundant: ACDF->EF->E

General algorithm for obtaining a minimal cover set

1. Put the FDs in a standard form (single attribute on the right)
2. Minimize the left side of each FD
3. Delete redundant FDs

Example 4: 3NF synthesis

- Schema: <u>C</u>SJDPQV
- FDs: {SD->P, J->S, JP->C, C->CSJDPQV}

1. Find the minimal cover set

Example 4: 3NF synthesis

- Schema: <u>C</u>SJDPQV
- FDs: {SD->P, J->S, JP->C, C->CSJDPQV}

1. Find the minimal cover set

   C->S, C->J, C->D, C->P, C->Q, C->V, SD->P, J->S, JP->C

Example 4: 3NF synthesis

- Schema: CSJDPQV
- FDs: {SD->P, J->S, JP->C, C->CSJDPQV}

1. Find the minimal cover set

~~C->S~~, C->J, C->D, ~~C->P~~, C->Q, C->V, SD->P, J->S, JP->C

C->S,C->D, SD->P => C->P (eliminated)
C->J, J->S => C->S (eliminated)

Example 4: 3NF synthesis

- Schema: <u>C</u>SJDPQV
- FDs: {SD->P, J->S, JP->C, C->CSJDPQV}

1. Find the minimal cover set

   ~~C->S~~, C->J, C->D, ~~C->P~~, C->Q, C->V, SD->P, J->S, JP->C

   2. Schemas: CJ, CD, CQ, CV, SDP, JS, JPC
   3. (Optional) Combine: CJDQVP, SDP, JS
   4. CJDQVP is a superkey
   5. Done

Case Study: The Internet shop

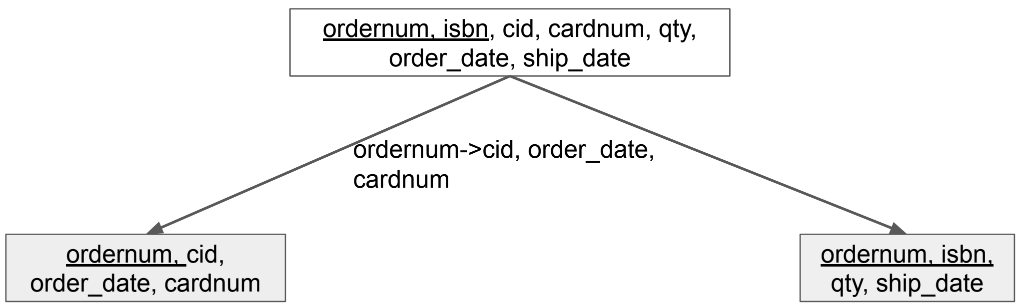
- Books (isbn: VARCHAR(10), title: VARCHAR(8), author: VARCHAR(80), qty_in_stock: int, price: double, year_published: int)
- Customers (cid: int, cname:VARCHAR(80), address: VARCHAR(200))
- Orders (ordernum: int, isbn: VARCHAR(10), cid: int, cardnum: VARCHAR(16), qty: int, order_date: DATE, ship_date: DATE)

Functional dependencies

- Books has one key: isbn. No other FDs
- Customers has one key: cid. No other FDs
- Orders has key: (ordernum, isbn). ordernum->cid, ordernum->order_date, ordernum->cardnum

Decomposition

- Orders (ordernum: int, isbn: VARCHAR(10), cid: int, cardnum: VARCHAR(16), qty: int, order_date: DATE, ship_date: DATE)
- ordernum->cid, ordernum->order_date, ordernum->cardnum



Decomposition

- Orders (<u>ordernum: int, isbn: VARCHAR(10)</u>, cid: int, cardnum: VARCHAR(16), qty: int, order_date: DATA, ship_date: DATE)
- ordernum->cid, ordernum->order_date, ordernum->cardnum
- Orders (<u>ordernum</u>, cid, order_date, cardnum)
- Orderlists (<u>ordernum, isbn</u>, qty, ship_date)

References

1. "The cow book": Database management systems by Raghu Ramakrishnan and Johannes Gehrke